International Journal of Informatics and Communication Technology (IJ-ICT)
Vol. 14, No. 2, August 2025, pp. 565~574
ISSN: 2252-8776, DOI: 10.11591/ijict.v99i1.pp565-574

Incremental prioritization using an iterative model for small-scale systems

Ameen Shaheen, Wael Alzyadat, Aysh Alhroob, A. Nasser Asfour
Department of Software Engineering, Faculty of Science and Information Technology, Al-Zaytoonah University of Jordan, Amman, Jordan

Article Info

Article history:

Received Sep 16, 2024
Revised Nov 21, 2024
Accepted Dec 15, 2024

Keywords:

Incremental model
Iterative model
SDLC models
Software development
Software engineering

ABSTRACT

To improve customer satisfaction during the requirement engineering process and create higher consistency in the developed software, there is a growing trend toward the development and delivery of software in an incremental manner. This paper introduces a novel approach to prioritizing the initial development of core subsystems. This prioritization ensures that the most critical subsystems, which contribute significantly to the project's overall success, are addressed first. Our method involves employing an incremental model with iterative modeling, where each subsystem is assigned a profitability score ranging from 1 to 10. The iterative model is then utilized to identify the most suitable subsystem for the next development stage. The results of our study indicate that utilizing the total profit weight in conjunction with the iterative model effectively identifies the central subsystem of the entire project. This approach proves to be the optimal starting point for development, helping streamline the process and contribute to a more efficient software delivery strategy.

This is an open access article under the CC BY-SA license.

Corresponding Author:

Ameen Shaheen
Department of Software Engineering, Faculty of Science and Information Technology
Al-Zaytoonah University of Jordan
Airport street, Amman, Jordan
Email: a.shaheen@zuj.edu.jo

1. INTRODUCTION

The software development approach known as "incremental development" involves dividing a system or project into smaller, more manageable segments or increments. Each of these increments is developed and delivered individually, with the functionality of each increment building upon that of the previous one [1], [2]. This approach allows for the regular release of working software, making it easier to gather feedback, adapt to changing requirements, and ensure that the project stays on track [3]. Incremental development is commonly employed in agile methodologies such as scrum, and it differs from traditional "waterfall" development in that it involves completing the entire project in a linear sequence [4], [5].

The incremental model is like divide and conquer methodology, where it can be used to build large systems [6]. So, when we have a large and complex problem then the incremental model involves breaking it down into smaller, more manageable components [7]. The incremental model is viewed as a modification to the waterfall model [8], as the software increases in size, sometimes we necessary to develop a large project, so it is easier to subdivide it into smaller components, this can make every component work standalone [9]. Developing systems through incremental release requires first providing essential operating functions, and then providing system users with improved and more capable versions of a system at regular intervals [10]. In

the incremental model, the component was developed in an overlapping fashion as shown in Figure 1, then all components had to be integrated and tested.

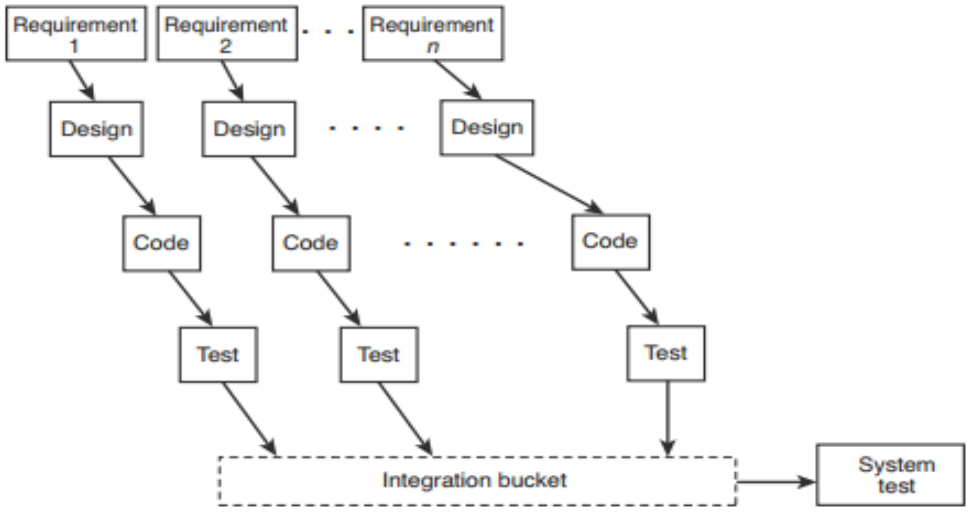

Figure 1. Incremental model

Iterative development is a crucial part of the software development industry, and it shows just how committed the industry is to innovation and adaptability [11]. It's important to embrace change, prioritize collaboration, and break projects into smaller pieces to ensure that software development teams can deliver top-quality software that meets the needs of users and stakeholders [12]. This approach is especially important given the constantly changing landscape of the digital age. By following the principles of iterative development, software development teams can navigate this complex terrain and achieve success [13].

Iterative development is also an approach that places significant emphasis on incremental progress, frequent feedback, and continuous improvement [14]. This approach recognizes the inherent complexity of software projects and acknowledges that requirements may change, evolve, or become more refined as the project moves forward. Rather than attempting to predict and plan for all of these changes in advance, iterative development divides the project into smaller, more manageable pieces known as iterations or cycles [15]. Each iteration encompasses the entire software development lifecycle, including planning, design, coding, testing, and evaluation as shown in Figure 2.

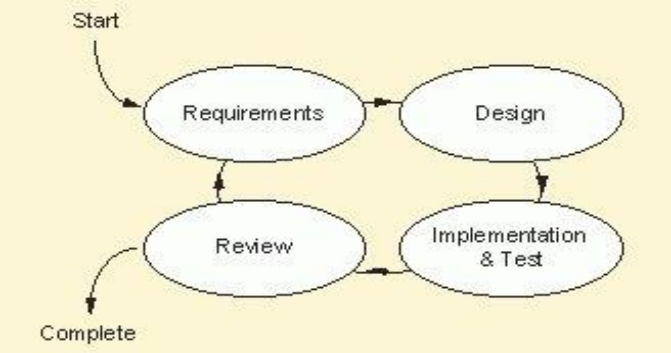

Figure 2. Iterative development

Prioritizing requirements is crucial for successful project management. By focusing on the most important elements first, teams can work in sync with business needs and ensure that critical elements are developed promptly [16]. The incremental model, which divides large systems into smaller subsystems, allows

for stand-alone work on each subsystem and facilitates iterative development. This approach emphasizes incremental progress, frequent feedback, and continuous improvement, recognizing the inherent complexity of software projects and the potential for requirements to change or evolve over time [17].

Establishing priorities for subsystems is a crucial task, as it can be difficult to determine which subsystem to address initially. This affords the opportunity to channel our efforts and resources effectively, resulting in progress. A thorough examination of each subsystem is necessary to decide on the appropriate course of action. Thus, it becomes imperative to establish priorities for each of them [18]. Prioritizing requirements is feasible by considering various aspects, including importance, cost, penalty, time, risk, and dependencies [19].

In this study, the research gap lies in the absence of effective methodologies for prioritizing subsystems throughout the early stages of incremental software development, especially in the demands of engineering process. Current methods frequently ignore the integration of profitability as a factor for prioritization, resulting in inefficient development sequences and dissimilar software outcomes. This study covers these gaps by offering a profitability-based iterative model that improves the process of decision-making, providing a more structured and result-oriented approach to software development.

2. RELEATED WORK

The literature extensively discusses various prioritization strategies for requirements, among which the analytical hierarchy process (AHP) emerges as the predominant priority-based technique. AHP is specifically designed to enable decision-makers to establish priorities and make informed decisions [20]. An additional technique is the bubble sort [21], wherein the underlying principal shares similarities with AHP. Both methods employ a pair-wise comparison operation. Another technique is the binary search tree where its establishment tree involves representing each requirement within a node. The prioritization of the tree entails situating low-priority requirements on the left side and high-priority requirements on the right [22]. While this method is efficient, the comparison inherent in the binary search tree is conventional, merely indicating the relative desirability of requirements without providing nuanced distinctions.

Cumulative voting, also known as the 100-dollar test, constitutes a straightforward process wherein stakeholders of the system are allocated 100 units for distribution among requirements. The prioritization in this method is determined by assigning higher units to requirements with higher priority and lower units to those with lower priority, with stakeholders controlling the distribution process. However, when dealing with a substantial number of requirements, this approach has drawbacks as it may not effectively calculate prioritization and could yield inaccurate results. Additionally, determining the appropriate quantity of units to allocate and the remaining units may pose challenges [23].

The spanning trees technique bears similarity to AHP in that both methodologies employ pair-wise comparison operations, but the former employs the minimum spanning technique. This is achieved through the utilization of a spanning tree architecture to eliminate redundant comparisons, thereby reducing the overall number of comparisons [24]. However, its efficiency diminishes when confronted with a large number of requirements.

Numerical assignment, involving the grouping of requirements, establishes a scale by categorizing them into distinct groups. Subsequently, each requirement is assigned a 5-point scale to evaluate its significance. However, it is noteworthy that this technique exhibits a diminished level of reliability and fault tolerance.

The Wieger method establishes the priority of a requirement by dividing its value by the combined costs and technological risks associated with implementation. Additionally, the method involves assessing customer significance through a 1-9 scale, along with considering the implications if the requirement were not implemented. Despite its utility, a drawback of the Wieger method lies in its susceptibility to stakeholder influence, potentially allowing stakeholders to manipulate the prioritization process to align with their specific objectives [25].

The MoSCoW technique centers on collaborative efforts between analysts and stakeholders to categorize requirements into four distinct groups. While its efficiency is notable, the approach necessitates human involvement, leading to potential disagreements between analysts and stakeholders. Consequently, this method is characterized by relatively low scalability, prompting exploration of other hybrid techniques to address its limitations [26].

3. INCREMENT PRIORITIZATION WITH ITERATIVE MODEL FOR SMALL SYSTEMS

Initiating the development of a project using the incremental model involves the segmentation of the project into subsystems. Each subsystem is designed to work independently, thereby necessitating the implementation of the software development life cycle (SDLC) for its development. In this context, the

incremental software development life cycle is employed. Furthermore, the iterative model is utilized to discern the priority sequence for initiating or proceeding with the development of each subsystem.

The challenge arises in determining the priority of each subsystem, particularly due to the inherent ambiguity within certain requirements. This ambiguity introduces the potential for stakeholders or developers to alter the prioritization. To mitigate this, the adoption of a combined iterative and incremental model is implemented. This strategic approach ensures a continual reassessment of priorities, thereby providing a heightened level of confidence in determining the optimal sequencing for the implementation of each subsystem.

Presently, the task at hand involves the selection of the most pivotal component with which to commence. To achieve this, a comprehensive assessment of the benefits associated with each subsystem is imperative, contingent upon several factors. It is noteworthy, however, that identifying these factors in a universally applicable manner for all systems proves to be a complex endeavor. In the scope of this paper, our focus is directed towards smaller systems wherein the factors outlined can be effectively implemented. These factors encompass:

1) Customer and contractual requirements benefit (CCRB):
   After dividing the system into subsystems, it becomes imperative to compose concise descriptions for each. Subsequently, stakeholders are solicited to provide evaluative values ranging from 1 to 10 for each respective subsystem.
2) Integration requirements (IR):
   In this context, stakeholders are invited to assess the integration among subsystems by assigning a numerical value ranging from 1 to 10 to each.
3) Time of the subsystem (T):
   Furthermore, stakeholders have the opportunity to appraise the required time for the construction of each subsystem by assigning numerical values within the range of 1 to 10.
4) Budget of the subsystem (B):
   Stakeholders have the authority to determine the costs associated with constructing each subsystem, assigning values within the range of 1 to 10 for each.
5) Similar type of project (S):
   This determination draws upon the collective experience and background of the stakeholders.

A stakeholder in the architectural framework of a system refers to an individual, team, organization, or relevant entity with a vested interest in the actualization of the system. The majority of system development projects typically involve participation from representatives of various stakeholder groups, with the extent of their significance varying across projects. Nonetheless, it is crucial to acknowledge each class of stakeholders, as neglecting any may lead to future complications. Striking a balance and establishing priorities among the diverse needs of stakeholder groups is essential. This ensures that when conflicts arise, informed and rational decisions can be made. In our scenario, stakeholders are categorized based on their respective roles and concerns, as shown in Table 1. However, when stakeholders do not possess equal weight within the project, an analytical technique is employed to discern the respective roles and weights attributed to each stakeholder. This determination is derived from an importance-influence matrix as shown in Figure 3.

Table 1. Stakeholders' categorization

Stakeholder	Role
Developers (D)	Develop and implement the system based on specifications, or take a leadership role in guiding the teams responsible for this process.
Production engineers (PE)	Design, deploy, and manage the system.
Project customer (PC)	The individual or entity requiring the system for their organization or company.
Users (U)	Specify the functional attributes of the system and, ultimately, engage in its utilization.

The conventional approach to stakeholder analysis involves the utilization of a stakeholder matrix. This matrix juxtaposes stakeholders along two distinct variables. These variables may encompass the degree of 'stake' in the project outcomes as opposed to the 'resources' vested in the stakeholder. Alternatively, it may involve assessing the 'importance' of the stakeholder relative to their 'influence'. While the conceptual framework remains consistent, the emphasis may vary slightly.

Boxes A, B, and C represent the principal stakeholders of the project. The implications associated with each box are summarized and outlined as follows:

– Box A
  These stakeholders exhibit a pronounced level of influence on the project and concurrently hold significant importance for its success. Consequently, the implementing organization is compelled to cultivate robust

working relationships with these stakeholders, thereby securing an influential coalition of support for the project. Illustrative examples could include senior officials, politicians, or trade unions.

- Box B
  These stakeholders hold significant importance in determining the success of the project, albeit with a relatively lower degree of influence. Consequently, safeguarding their interests necessitates the implementation of targeted initiatives. An illustrative example could encompass traditionally marginalized groups, such as Indigenous people, youth, and seniors, who may stand to benefit from a new service but possess limited influence in its development.
- Box C
  These stakeholders wield substantial influence, thus possessing the capacity to impact the project outcomes; however, their interests may not inherently align with the overarching goals of the project. Noteworthy examples include financial administrators, who hold significant discretionary powers over funding disbursements. This assertion suggests that these stakeholders represent a noteworthy source of risk, demanding meticulous monitoring and strategic management.
- Box D
  Stakeholders categorized within this designation, characterized by minimal influence or significance concerning the project objectives, may necessitate only modest levels of monitoring or evaluation and are considered of lower priority.

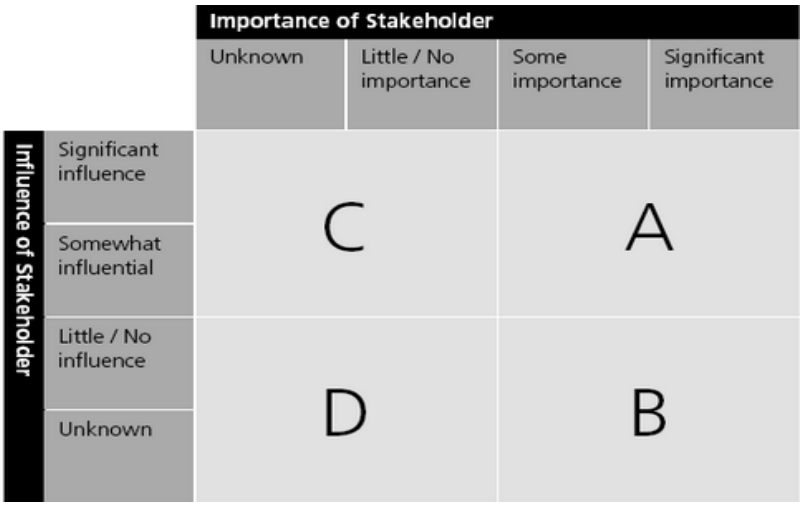

Figure 3. Importance-influence matrix

Within this paper, stakeholders are assigned a maximum range value determined by their placement in the importance–influence matrix, guided by our specified factors. The details are outlined in Table 2. According to the Table 2, stakeholders can be invited to provide their input through a voting mechanism with fine granularity. For example, following a collaborative brainstorming session among all stakeholders to categorize them within the importance–influence matrix for each factor, the resultant allocation is expected to be as shown in Figure 4.

Table 2. Maximum range value

Box	Max value
A	4
B	3
C	2
D	1

Following the establishment of an importance–influence matrix for each factor, stakeholders can be invited to participate in a voting process within their designated range. For instance, the outcome may manifest as shown in Table 3.

It is noteworthy to observe that there may be instances where multiple stakeholders fall within the same voting range. Subsequent to the voting process, the foundational increment, from which we can commence, can be deduced as (1).

$$Total\ Profit = CCRB + IR + T + B + S \quad (1)$$

However, it is imperative to exercise caution regarding stakeholder voting, as each stakeholder category may encompass multiple individuals. Therefore, each constituent within a category must cast their vote individually. Subsequently, the summation of all individual votes is necessary. For instance, in the case of multiple developers, the summation would be computed as (2).

$$Developer\ voting = \sum CCRB + IR + T + B + S \quad (2)$$

Subsequently, following the aggregation of votes from all stakeholders, the computation of the overall profit for each increment becomes feasible as (3).

$$Total\ Profit = \sum CCRB + \sum IR + \sum T + \sum B + \sum S \quad (3)$$

The outcome derived from (3) is expected to be a numerical value. The subsystem associated with the highest numerical value is designated as the starting point, while the remaining increments are to be systematically organized in descending order within a queue as shown in Figure 5.

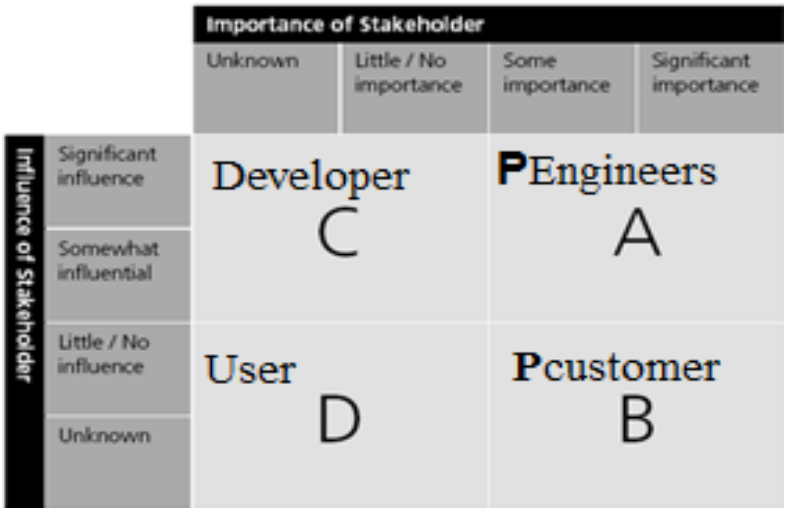

Figure 4. Voting mechanism

Table 3. Importance influence matrix for each factor

	CCRB	IR	T	B	S
D	0-2	0-3	0-3	0-2	0-4
PE	0-2	0-4	0-3	0-4	0-3
PC	0-4	0-2	0-4	0-4	0-2
U	0-4	0-1	0-1	0-1	0-1

By employing this formalism, we can delineate the subsystems suitable for initial development. However, this process may introduce some ambiguity during the construction of the initial subsystem. This ambiguity arises due to:

i) Other subsystems may exhibit greater clarity for the stakeholders.
ii) Alterations to requirements may occur.
iii) Lack of prior experience, especially in the context of a new project.
iv) New technology or platform.

This necessitates each stakeholder to modify the assigned values accordingly.

Hence, subsequent to the development of the initial increment and its delivery to the stakeholders, the iterative model will be employed to mitigate any ambiguity in the factors determining the next subsystem for development. Consequently, each stakeholder is required to provide a new value for each of the remaining subsystems. This procedural sequence can be instantiated as algorithm shown in Figure 6.

Through the iterative algorithm, recalculations are performed to derive the new profit for each subsystem. The subsystem exhibiting the highest profit is subsequently assigned for development by the developer team, leading to a decrement in the total number of subsystems by 1. Despite the additional time investment incurred by the utilization of the iterative model, its implementation yields the advantage of ensuring the selection of the optimal subsystem for subsequent development. This is attributed to the enhanced clarity gained by all stakeholders regarding the factors involved, especially after the initial subsystem has been developed.

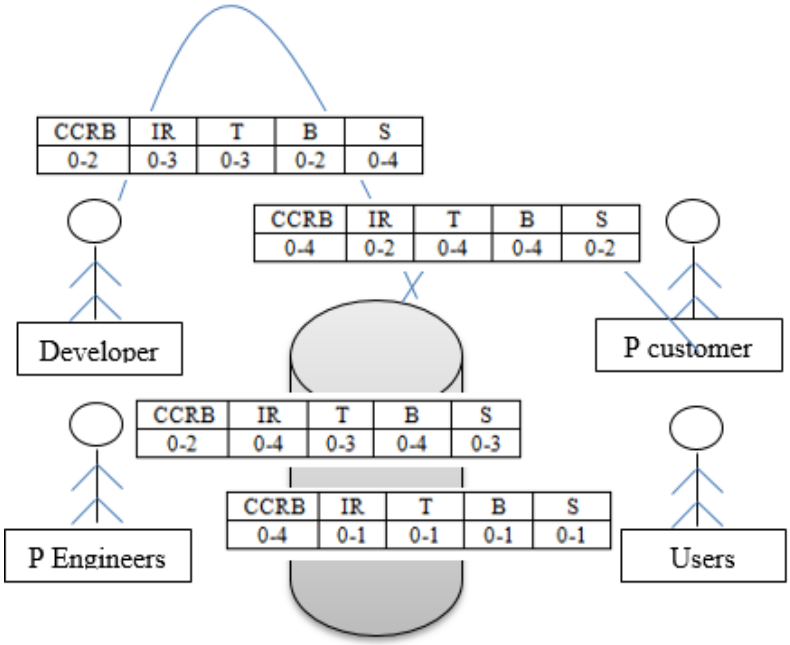

Figure 5. First increment result

```
Input: TProfit for each Increment – First one. OldTp []
Output: New queue Content new Arranged. NewTP []
Count= Number of existing Increment
For all Count
   NewTP[i]=∑ (CCRB+IR+T+B +S)
Sort (NewTP[i])
If Max (NewTP[i])> Max (OldTP[i])
Select the Increment (Max (NewTP[i]))
Else
Select the Increment (Max (OldTP[i]))
Count --
```

Figure 6. Iterative algorithm

4. CONCLUSION

In conclusion, this research introduces a novel approach to prioritizing core subsystems in the incremental software development process by utilizing the total profit weight in conjunction with the iterative model facilitates the determination of the central subsystem within the entire project, serving as the optimal starting point for development. This approach aids engineers and developers in identifying the project components that merit the highest priority, directing the majority of their attention accordingly. The proposed model not only discusses the issues of inconsistency in traditional software development models, but it also

assures that the most crucial and profitable subsystems are developed early on, in accordance with both business and customer requirements. The results illustrate that this approach has enormous potential to improve the software development process by providing a more dynamic and flexible methodology that can respond to changing project needs and market demands.

ACKNOWLEDGEMENTS

We would like to express our deepest gratitude to everyone who contributed to the success of this project, titled "Development and evaluation of the effectiveness of the electronic donation management system for the Jordanian Food Bank using artificial intelligence techniques", Project Number: 13 / 16 / 2023-2024.

We would like to thank also Al-Zaytoonah University of Jordan for providing us with the opportunity and the Jordanian Food Bank for the necessary resources to undertake this project. The support from the administration and the IT department has been invaluable in every step of our research and development process.

FUNDING INFORMATION

This research was funded by Al-Zaytoonah University of Jordan under the project titled "Development and Evaluation of the Effectiveness of the Electronic Donation Management System for the Jordanian Food Bank Using Artificial Intelligence Techniques", Project Number: 13 / 16 / 2023-2024. The authors acknowledge the Jordanian Food Bank for providing the necessary resources to undertake this project.

AUTHOR CONTRIBUTIONS STATEMENT

Name of Author	C	M	So	Va	Fo	I	R	D	O	E	Vi	Su	P	Fu
Ameen Shaheen	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	
Wael Alzyadat	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	
Aysh Alhroob	✓		✓	✓	✓	✓	✓		✓	✓		✓	✓	
A. Nasser Asfour	✓		✓	✓		✓	✓		✓	✓	✓		✓	

C : **C**onceptualization
M : **M**ethodology
So : **So**ftware
Va : **Va**lidation
Fo : **Fo**rmal analysis
I : **I**nvestigation
R : **R**esources
D : **D**ata Curation
O : Writing - **O**riginal Draft
E : Writing - Review & **E**diting
Vi : **Vi**sualization
Su : **Su**pervision
P : **P**roject administration
Fu : **Fu**nding acquisition

CONFLICT OF INTEREST STATEMENT

The authors declare that they have no known competing financial interests or personal relationships that could have influenced the work reported in this paper.

DATA AVAILABILITY STATEMENT

The data that support the findings of this study are available upon request from the corresponding author, Ameen Shaheen. Due to institutional and privacy regulations, the data are not publicly available.

REFERENCES

[1] R. Hoda, N. Salleh, J. Grundy, and H. M. Tee, "Systematic literature reviews in agile software development: a tertiary study," *Information and Software Technology*, vol. 85, pp. 60–70, May 2017, doi: 10.1016/j.infsof.2017.01.007.

[2] M. Muhairat, W. Alzyadat, A. Shaheen, A. Alhroob, and A. N. Asfour, "Leveraging machine learning for predictive pathways in higher education: a case study at Al-Zaytoonah University of Jordan," *International Journal of Electronics and Communication Engineering (IJECE)*, vol. 11, no. 11, pp. 28–44, Nov. 2024, doi: https://doi.org/10.14445/23488549/ijece-v11i11p104.

[3] J. A. Highsmith, "Adaptive software development: a collaborative approach to managing complex systems," *Journal of Evolutionary Biology*, vol. 12, p. 392, 2000, [Online]. Available: http://books.google.com/books?id=R1ZyQgAACAAJ.

[4] R. Shaydulin and J. Sybrandt, "To Agile, or not to agile: a comparison of software development methodologies," *Arxiv*, 2017, [Online]. Available: http://arxiv.org/abs/1704.07469.

[5] C. Chen, H. A. Jacobsen, and R. Vitenberg, "Algorithms based on divide and conquer for topic-based publish/subscribe overlay design," *IEEE/ACM Transactions on Networking*, vol. 24, no. 1, pp. 422–436, Feb. 2016, doi: 10.1109/TNET.2014.2369346.
[6] M. H. Altarawneh, W. Alzyadat, B. M. Alwadi, A. Al-Shaikh, A. Shaheen, and A. Alhroob, "The relationship between cross-cutting factors and knowledge, learning outcomes, and skills in dual degree programs," *Journal of Theoretical and Applied Information Technology*, vol. 102, no. 8, pp. 3410–3422, 2024.
[7] Y. Al-Kasabera, W. Alzyadat, A. Alhroob, S. Al Showarah, and A. Thunibat, "An automated approach to validate requirements specification," *Compusoft*, vol. 9, no. 2, pp. 3578–3585, 2020.
[8] S. Pargaonkar, "A comprehensive research analysis of software development life cycle (SDLC) Agile & waterfall model advantages, disadvantages, and application suitability in software quality engineering," *International Journal of Scientific and Research Publications*, vol. 13, no. 8, pp. 120–124, Aug. 2023, doi: 10.29322/IJSRP.13.08.2023.p14015.
[9] M. Lafi, B. Hawashin, and S. AlZu'bi, "Eliciting requirements from Stakeholders' responses using natural language processing," *CMES - Computer Modeling in Engineering and Sciences*, vol. 127, no. 1, pp. 99–116, 2021, doi: 10.32604/cmes.2021.013026.
[10] A. M. Almanaseer, W. Alzyadat, M. Muhairat, S. Al-Showarah, and A. Alhroob, "A proposed model for eliminating nonfunctional requirements in Agile methods using natural language processes," in *2022 International Conference on Emerging Trends in Computing and Engineering Applications, ETCEA 2022 - Proceedings*, Nov. 2022, pp. 1–7, doi: 10.1109/ETCEA57049.2022.10009796.
[11] N. Salleh, E. Al-Kautsar, R. Hoda, and A. liza Asmawi, "A window into the emergence of agile software development landscape in Indonesia," *International Journal of Advances in Soft Computing and its Applications*, vol. 6, no. 1, pp. 1–17, 2014.
[12] Q. Li, Y. Yang, M. Li, Q. Wang, B. W. Boehm, and C. Hu, "Improving software testing process: feature prioritization to make winners of success-critical stakeholders," *Journal of software: Evolution and Process*, vol. 24, no. 7, pp. 783–801, Nov. 2012, doi: 10.1002/smr.512.
[13] A. Ojeniyi, A. Khalid, and O. C. Kehinde, "A persuasive agent architecture for behavior change intervention," *International Journal of Informatics and Communication Technology (IJ-ICT)*, vol. 11, no. 2, p. 128, Aug. 2022, doi: 10.11591/ijict.v11i2.pp128-139.
[14] A. M. Hassan and A. John, "Comparative analysis on different software piracy prevention techniques," *International Journal of Informatics and Communication Technology (IJ-ICT)*, vol. 10, no. 1, p. 1, Apr. 2021, doi: 10.11591/ijict.v10i1.pp1-8.
[15] A. Al-Shaikh, A. Shaheen, M. R. Al-Mousa, K. Alqawasmi, A. S. Al Sherideh, and H. Khattab, "A comparative study on the performance of 64-bit ARM processors," *International Journal of Interactive Mobile Technologies*, vol. 17, no. 13, pp. 94–113, Jul. 2023, doi: 10.3991/ijim.v17i13.39395.
[16] R. Morien, "An agile software project management manifesto – a reference disciplines framework for agile development," *International Journal of Soft Computing and Its Applications (IJASCA)*, vol. 6, no. 1, pp. 1–19, 2014.
[17] I. Scoones *et al.*, "Transformations to sustainability: combining structural, systemic and enabling approaches," *Current Opinion in Environmental Sustainability*, vol. 42, pp. 65–75, Feb. 2020, doi: 10.1016/j.cosust.2019.12.004.
[18] S. Gautam, D. Kumar, and L. M. Patnaik, "Neuro-fuzzy approach for software release time optimization," *International Journal of Advances in Soft Computing and its Applications*, vol. 9, no. 3, pp. 36–48, 2017.
[19] R. Thakurta, "Understanding requirement prioritization artifacts: a systematic mapping study," *Requirements Engineering*, vol. 22, no. 4, pp. 491–526, Nov. 2017, doi: 10.1007/s00766-016-0253-7.
[20] V. Omkarprasad and K. Sushil, "Analytic hierarchy process: an overview of applications," *European Journal of operational research*, vol. 169, no. 1, pp. 1–29, 2006.
[21] M. Lafi and A. A. Qader, "A novel automated requirements prioritization and selection model based on requirements weights and stakeholder importance," *International Review on Computers and Software (IRECOS)*, vol. 18, no. 1, p. 44, Jun. 2023, doi: 10.15866/irecos.v18i1.23246.
[22] K. M. Daud *et al.*, "A hybrid of differential search algorithm and flux balance analysis to: identify knockout strategies for in silico optimization of metabolites production," *International Journal of Advances in Soft Computing and its Applications*, vol. 10, no. 2, pp. 84–107, 2018.
[23] K. Riņķevičs and R. Torkar, "Equality in cumulative voting: a systematic review with an improvement proposal," *Information and Software Technology*, vol. 55, no. 2, pp. 267–287, Feb. 2013, doi: 10.1016/j.infsof.2012.08.004.
[24] W. Alzyadat, A. Shaheen, A. Al-Shaikh, A. Alhroob, and Z. Al-Khasawneh, "A proposed model for enhancing e-bank transactions: an experimental comparative study," *Indonesian Journal of Electrical Engineering and Computer Science*, vol. 34, no. 2, pp. 1268–1279, May 2024, doi: 10.11591/ijeecs.v34.i2.pp1268-1279.
[25] I. Ibriwesh, S. B. Ho, I. Chai, and C. H. Tan, "Prioritizing solution-oriented software requirements using the multiple perspective prioritization technique algorithm: an empirical investigation," *Concurrent Engineering Research and Applications*, vol. 27, no. 1, pp. 68–79, Mar. 2019, doi: 10.1177/1063293X18808559.
[26] T. Kravchenko, T. Bogdanova, and T. Shevgunov, "Ranking requirements using MoSCoW methodology in practice," in *Lecture Notes in Networks and Systems*, vol. 503 LNNS, 2022, pp. 188–199.

BIOGRAPHIES OF AUTHORS

Ameen Shaheen received his B.Sc. in computer information system in 2010, and M.S. in computer information system in 2012, and Ph.D. in computer science in 2018. He currently works as assistant professor at the Software Engineering Department, at Al-Zaytoonah University of Jordan. His expertise is in database, algorithms, and systems analysis. He can be contacted at email: a.shaheen@zuj.edu.jo.

Wael Alzyadat is associate professor who works at the Software Engineering Department, Al-Zaytoonah University of Jordan. The ongoing projects are the development of a new big data model for decision making: case study -drugs chemical structures and big data coherence with KDD. He can be contacted at email: wael.alzyadat@zuj.edu.jo.

Aysh Alhroob currently works as a Dean, the Faculty of Science and Information Technology and a full professor at the Department of Software Engineering, Al-Zaytoonah University, Jordan. He has done his Ph.D. at the University of Bradford, UK. His expertise is in data and text mining, big data analysis, software testing, and software requirements. He can be contacted at email: aysh@zuj.edu.jo.

A. Nasser Asfour is assistant professor who works at the Software Engineering Department, at Al-Zaytoonah University of Jordan. He had done his Ph.D. at Politehnica University of Bucharest, Romania. His expertise is in data and text mining, database, and systems analysis. He can be contacted at email: a.nasser.asfour@zuj.edu.jo.